The Driving School System: Learning Automated Basic Driving Skills from a Teacher in a Real Car

Irene Markelić, Anders Kjær-Nielsen, Karl Pauwels, Lars Baunegaard With Jensen, Nikolay Chumerin, Aušra Vidugiriene, Minija Tamosiunaite, Alexander Rotter, Marc Van Hulle, Norbert Krüger, and Florentin Wörgötter

***Abstract*—We present a system that learns basic vision based driving skills from a human teacher. In contrast to much other work in this area which is based on simulation, or data obtained from simulation, our system is implemented as a multi-threaded, parallel CPU/GPU architecture in a real car and trained with real driving data to generate steering and acceleration control for road following. In addition it uses a novel algorithm for detecting independently moving objects (IMOs) for spotting obstacles. Both, learning and IMO detection algorithms, are data driven and thus improve above the limitations of model based approaches. The system's ability to imitate the teacher's behavior is analyzed on known and unknown streets and the results suggest its use for steering assistance but limit the use of the acceleration signal to curve negotiation. We propose that this ability to adapt to the driver has high potential for future intelligent driver assistance systems since it can serve to increase the driver's security as well as the comfort, an important sales argument in the car industry.**

***Index Terms*—imitation learning, driving, real-time system, independently moving object, advanced individualized driver assistance system**

ADVANCED driver assistance systems (ADAS) that adapt to the individual driver have a high potential for the car industry since they can reduce the risk of accidents while providing a high degree of comfort. Conventional systems are based on a general moment to moment assessment of road and driving parameters. To arrive at a judgment of the current situation they use control laws from which they derive output signals to aid the driver [1]–[4]. However, inter-individual differences in driving can be large, e.g., it is known that different people pursue different driving styles, [5], and driving strategies [6]. It is difficult for conventional control systems to accommodate these differences which leads to suboptimal driver support. However, this can have negative effects on the driver's behavior [7], [8], i.e., safety is decreased instead of increased. Observations like these were the motivation for us to investigate and build a system that drives in the same way as its user(s). These reasons also lead to the situation that current R&D efforts of the car industry focus on systems which take the driver and its behavior explicitly into account [9], [10]. In addition to the safety aspect, such systems will also be accepted more easily by the user because they will provide more comfort; an important sales argument.

In the current study we will describe a system based on imitation learning, hence, a system that learns to interpret basic aspects of the road (lanes) in the same way as its driver, reproducing the driver's actions. In addition we demonstrate its use as a basic driver assistance system by issuing warning signals if the driver deviates from his/her predicted default behavior. The so-called DRIVSCO[1] system is realized by a multi-threaded, parallel CPU/GPU architecture. It is vision based, operates in real-time on real roads and also includes a novel form of data driven detection of independently moving objects (IMOs). The system has been designed for the use on motorways and country roads.

Before describing the details of our system and comparing it to the literature (see State of the Art), we will shortly explain its structure as a guideline for the reader (Fig. 1). Thus, this paper is organized as follows: in Section I the overall structure of the system is presented. It is compared to the state of the art in Section II, its realization explained in Section III and results presented in Section IV. In Section V we conclude and discuss the presented work.

I. System Overview

The overall structure of the DRIVSCO system is shown in Fig. 1. The yellow box, "human sense-act", symbolizes the human driver who senses the environment, denoted by the "world" box, and responds to it with adequate driving actions, "act". At the same time the system senses the environment, "system sense", via recorded image frames. The sensing process consists of detecting left and right street lanes, "sense lane", and IMOs, "senseIMO". The latter refer to objects that move with respect to the static environment, where detection is not restricted to predefined objects such as other cars or motorcycles. If an IMO is detected a warning is triggered, "warning IMO", if it is considered to affect the driver (see Section III-D). Once lane detection is initialized it is compared

I. Markelić and F. Wörgötter are with the Institute of Physics 3, Georg-August-University Göttingen, Germany, Friedrich-Hund Platz 1, D-37077 Göttingen, Germany (e-mail: {irene, worgott}@physik3.gwdg.de).)

A. Kjær-Nielsen, L. Baunegaard With Jensen and N. Krüger are with The Maersk Mc-Kinney Moller Institute, University of Southern Denmark, Campusvej 55, DK-5230 Odense M, Denmark (e-mail: {akn, lbwj, norbert}@mmmi.sdu.dk).

K. Pauwels, N. Chumerin and M. Van Hulle are with Katholieke Universiteit Leuven, Herestraat 49, bus 1021, BE-3000 Leuven, Belgium (e-mail: {Karl.Pauwels, Nikolay.Chumerin, Marc.VanHulle}@med.kuleuven.be).

A. Rotter is with Hella KGaA Hueck & Co, Rixbecker Str. 75, D-59552 Lippstadt, Germany, (e-mail: Alexander.Rotter@hella.com).

A. Vidugiriene and M. Tamosiunaite are with the Vytautas Magnus University, K.Donelaicio g. 58, LT-44248, Kaunas, Lithuania (e-mail: {m.tamosiunaite,a.vidugiriene}@if.vdu.lt).

Manuscript received March 12, 2010

This work has been supported by the European Commission under project FP6-IST-FET (DRIVSCO) and by the BFNT Göttingen.

[1]short for Driving School.

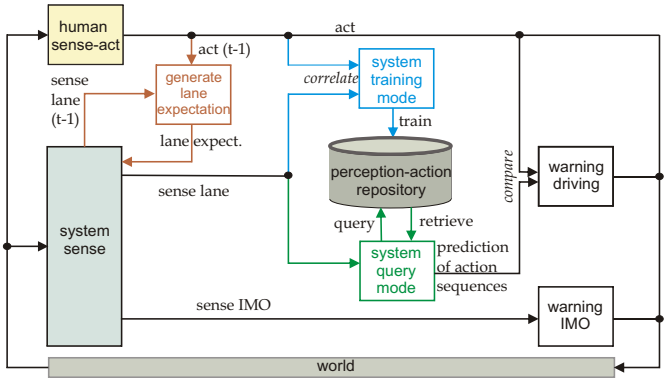


Fig. 1: Block diagram of the DRIVSCO overall system structure. The notation $t-1$ indicates an earlier time frame.

to a prediction about the expected lane positions, "generate lane expectation", which is obtained by using previously captured human action data and the previously detected lane, indicated by the notation $t-1$, to predict the current lane structure. This way incorrectly detected lanes can be filtered out (see Section III-E). The learning system is realized by the perception-action repository, PAR, depicted as the gray container in the figure, which serves as the system's memory where it can store its observations and retrieve them at a later stage (see Section III-F). The experience stored in the PAR is a state vector containing a concise lane description extracted from the current image frame and sequences of preceding and succeeding action data. Thus, there are two modi: 1) *training*, during which the system gathers experience to fill the PAR, denoted in blue "system training mode" (explained in Section III-F2), and 2) *retrieval* during which the system queries the PAR to retrieve stored knowledge denoted in green "system query mode" (see Section III-F3). Since the goal is to learn lane following, i.e., context-free driving, training the PAR requires the recorded data to be filtered to free it from context-dependent scenes, comp. Section III-F1. This makes the training phase a demanding procedure which is done off-line, i.e., not during driving. The retrieval mode however, can be used off- and online. Based on accumulated PAR returns, action plans for expected human steering and acceleration behavior for lane following are predicted, "prediction of action sequences" (see Section III-F4). To demonstrate the system's use as an assistance system the predicted action sequence is compared to the driver actions and a warning is issued if they are differing too much, "warning driving" (see Section III-F4). In other words, the driver is informed when deviating from his or her learned default driving behavior. The warning signals are displayed on a screen close to the driver, shown in Fig. 2c and 4.

II. State of the Art

Since our work aims at generating sequences of future driving actions based on visual sensory input it is closely related to vision based autonomous driving. Many approaches in this field rely on control theoretic white-box methodologies, i.e., they are based on predefined analytical models and control laws. This has led to some well performing systems, e.g., [11]–[14], however, drawbacks are the dependency on predefined knowledge, and the difficulty of developing models and control laws which restricts the design process to experts. In addition these systems do not, or only in a limited way, provide a means for individualization. By contrast, imitation learning [15] aims at extracting a policy for a given task by using examples provided by a teacher. This reduces the amount of required a priori knowledge and the need of explicit programming and thus facilitates human computer interaction. A famous example of imitation learning for driving is the ALVINN system [16]–[19] where actions of a human driver were associated with concurrent visual input from a camera via a neural network. The inputs to the network were the pixel values of the (downscaled) camera image and the output was a distribution of steering angles. Velocity control was handled by the driver. Further imitation learning work by Pasquier describes the learning of several driving skills with a fuzzy neural network [20]. Such motor skills are difficult to formalize and the fuzzy net represents a way of capturing semantic rules from continuous control input. It was aimed at emulating the role of the cerebellum during cognitive skill learning, and the algorithms for lane following were tested in simulation. Similar work, but dealing with helicopter flying, was reported in [21]. A novel form of inverse reinforcement learning [22] was introduced in [23] and applied to learning particular driving styles from example data obtained from simulation.

The EU funded project DIPLECS [24] reports similar goals to ours, however, DIPLECS does not aim at building a complete system. Research conducted at Motorola [25] aims at building an adaptive driver support system using machine learning tools. Its architecture was partially implemented in a prototype system built upon a simulator.

In addition to lateral control (steering), new generations of driver assistance systems will contain support for longitudinal control (velocity) during curve negotiation, e.g., [26]. Current, (non-imitation based) methods are usually not taking the street trajectory into account but simpler aspects like an obstacle in front (e.g., Adaptive Cruise Control systems using radar or laser for obstacle detection), known speed limits (e.g., Intelligent Speed Adapters and Limiters [27]) or leading vehicles (e.g., [28]). Focusing on curve negotiation and based on imitation learning are [20], [29] and [30], which all employ fuzzy neural networks trained on human control data.

Our work, unlike similar approaches, describes the realization of a complete system implemented in a real car that learns the prediction of action plans, i.e., sequences of steering and acceleration actions, together with a novel form of IMO detection. One big difference to others is that we use data obtained from real car driving and not from a simulator. (Most of the presented algorithms were first tested on a robot platform as reported in [31].) By contrast to the implicit mapping between sensory input and actions achieved with the neural network in the ALVINN project, our *lazy learning* approach [32] realized by the PAR allows the preservation of human interpretable information processing at all stages. Although this comes at the cost of having to store more data,

a b c

Fig. 2: System integration in the car. (a) Mounting for the stereo camera system. (b) Fixation of the computer in the trunk. (c) Installation of the monitor in the car.

it is highly beneficial concerning intuitive error analysis.

III. Submodules and Methods

A. Hardware

The used car is a Volkswagen Passat provided by the DRIVSCO partner Hella KGaA Hueck & Co (Lippstadt, Germany). The following sensory data is accessed via the CAN-bus: steering angle, indicating the steering wheel's position and taking values between $-360°$ (left steering) and $360°$ (right steering), velocity in km/h, longitudinal acceleration taking values between $-10\text{m}/s^2$ and $10\text{m}/s^2$, and curve radius measured by a gyroscope and taking values between -15000m and 15000m. Furthermore a camera stereo rig is mounted behind the windscreen as shown in Fig. 2a. We use two color Pulnix TM-1402CL cameras which deliver 1280×1024 raw Bayer pattern images at a frequency of 20 Hz. Furthermore a Global Positioning System (GPS) receiver was added to allow the visualization of driven routes, for which we used Google Maps, comp. Fig. 6. All computations are carried out on a PC with an Intel Core i7 - 975, 3.33 GHz quad–core processor with Simultaneous multi-threading (SMT) enabled and 12 GB RAM. The used graphics cards are an NVIDIA GTX295 for computation and a smaller one for displaying purposes. The PC is kept in the car's trunk as shown in Fig. 2b and the output of the system is shown on a screen next to the steering wheel as shown in Fig. 2c.

B. System Architecture

The backbone of this work is its realization as a multi-threaded, pipelined real-time system where shared memory is used for interprocess communication. Due to the complexity of the pipeline structure and the involvement of multiple CPU-cores and multiple GPUs in its computation, we have developed our own modular architecture. By simplifying the coordinated use of the heterogeneous computational resources in modern computers, this architecture allows for independent development of the individual processing stages. To our knowledge, no such CPU/GPU pipeline framework combining task- and data-parallelism exists, allowing a stage to share data with multiple other stages — e.g., the output of the preprocessing stage is used by both "lane detection" and "dense vision", as seen from Fig. 3. The system structure from Fig. 1 is realized by the architecture shown in Fig. 3. Each information processing entity is referred to as a stage and runs in a thread as indicated. All stages are connected through a

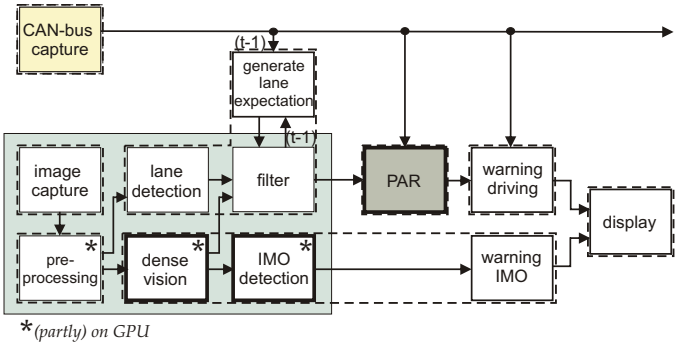


Fig. 3: The realization of the DRIVSCO system. Boxes denote processing stages and the dashed frames indicate individual parallel threads. Arrows denote the information flow, with the exception that the display stage connects to *all* stages. The "PAR", "dense vision" and "IMO detection" are key parts of this system. The notation $t-1$ indicates an earlier time frame.

pipeline where communication is realized by shared memory buffers to which processes can write to and read from. The processing time of each individual stage is below 50ms and the entire system works at the camera rate of 20 Hz. The "preprocessing", "dense vision" and "IMO detection" stages involve massive computations but achieve a frequency of 20 Hz through the use of two GPUs while all other processes run on the CPU. The "CAN-bus capture" stage is triggered every 50ms and makes the relevant car CAN-bus data available to other processes. During the "image capture" stage raw camera data is received and white level calculations are performed to control the camera's shutter time. The images are then undistorted, rectified and downsampled to 640×512 pixels during the "preprocessing". The boxes "lane detection", "dense vision" and "IMO detection" are explained in Section III-E, III-C, and III-D in detail. In addition a special display unit is integrated which connects to all buffers in the pipeline allowing the user to view the output of *any* stage, including the generated warning signals by means of a graphical user interface (GUI). A screenshot of this GUI is given in Fig. 4.

It shows a current image with detected lanes on the left, and a history and prediction of steering angles of two seconds on the right. The prediction is plotted in blue and the actual human steering in red. In addition the computed egomotion from the IMO detection stage is displayed in green. The gray dashed boundaries around the prediction indicate predefined threshold values used to issue a warning if exceeded by the actual driving. In this case the "unexpected driving" button below the displayed lane detection image is flashed. The system can also conveniently be used in an off-line mode to replay recorded scenes which is important for error analysis. Note, this display is designed for R&D purposes and should be simplified for a real driver assistance system.

C. Dense Vision

The visual cues used for the IMO detection are computed during the "dense vision" stage, see Fig. 3. Dense vision algorithms process the visual signal in its entirety, as opposed

Fig. 4: One tab of the system GUI, showing the detected lanes in the current image on the left and a history of prediction and steering angles of two seconds on the right along with the computed egomotion. Thresholds are shown in gray.

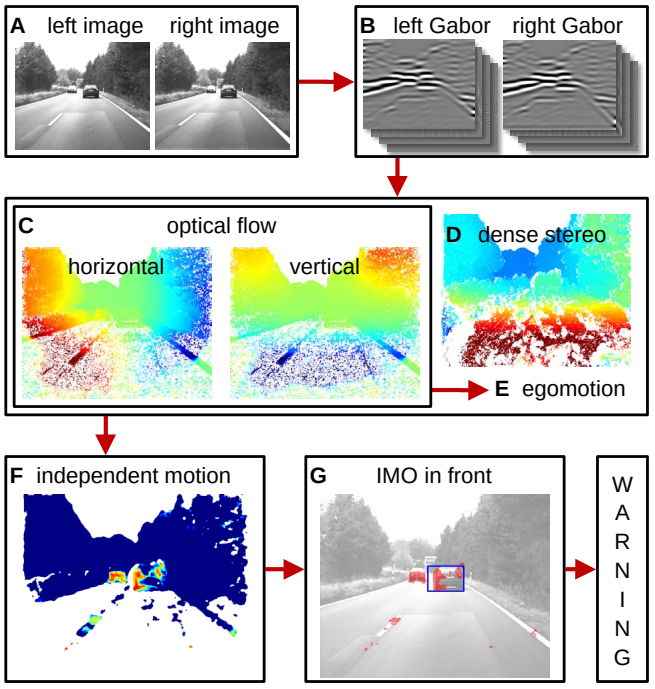


Fig. 5: Real-time IMO detection. Multi-orientation, multiscale Gabor filter responses (**B**) are extracted from the stereo image pair (**A**) and used to compute dense optical flow (**C**) and stereo disparity (**D**). The horizontal and vertical optical flow is color-coded from -15 (dark red) to +15 pixels (dark blue) and the stereo disparity from -50 (dark red) to +20 (dark blue) pixels. Combined with egomotion (**E**, extracted from the optical flow, not shown) these cues allow the extraction of independent motion (**F**, likelihood increases from blue to red). This independent motion signal is gathered in a fixed region of the image (**G**) and when it exceeds a threshold, a warning is issued.

to sparse vision algorithms that only operate on interest points such as edges or corners. The cues used here are dense disparity (the horizontal difference in image location for corresponding pixels in the rectified left and right image) and optical flow (the 2D image motion of each pixel). The algorithms used rely on the phase of quadrature pair Gabor filter responses [33] (Fig. 5B), extracted at multiple orientations and scales, to establish correspondences. The GPU implementation [34] of a phase-based algorithm [35] is used to compute optical flow (Fig. 5C). This algorithm integrates the temporal phase gradient across orientation, and gradually refines its estimates by traversing a Gabor pyramid from coarser to finer levels. The optical flow is computed for the left video stream only. The dense disparity algorithm (Fig. 5D) is very similar to the optical flow algorithm but operates on phase differences between the left and right image as opposed to temporal phase gradients. These aspects are described in more detail in [36].

D. IMO Detection and Warning

Recent (off-line) approaches for the detection of IMOs have achieved very good results using model-based techniques [37], [38]. One limitation of such approaches is the difficulty to detect IMOs early on. Distant objects will occupy only small patches in the image that are difficult to match to the model. Another limitation is that moving objects, for which no model is provided, cannot be detected and are simply ignored by the system. In this work we rely on a model-free detection mechanism that is more general in the sense that it will respond not only to cars, but to any sufficiently large ($11 \times 11 pixels$) moving object. The IMO detection component combines dense vision cues (optical flow and stereo, see Section III-C) in real-time to compute egomotion (the rotation and translation of the camera) and independent motion (the parts of the image that move with respect to the static environment) to detect an IMO in front.

The whole process is complex and cannot be described in detail in the current paper. Please refer to [39] for further information. Here, we will give only a short overview summarized in Fig. 5. A nonlinear instantaneous-time model [40] is used to extract egomotion from the optical flow. To obtain optimal estimates, an iterative minimization procedure is used that relies on M-estimation [41] for outlier compensation. A total of 32 different initializations are explored to deal with local minima. The data-intensive parts of the algorithm run entirely on the GPU. Independent motion is detected by evaluating the depth/flow constraint [42] at each pixel. Deviations from this constraint point to inconsistencies between the optical flow, disparity and egomotion and result from noise or independent motion. The deviations are assigned to independent motion if they comply with a 3D translation model in a local region surrounding the pixel (Fig. 5F). To detect a moving vehicle in front, the (pixelwise) independent motion signal is accumulated inside a fixed region in the image (blue rectangle in Fig. 5G). A warning is issued when more than 30% of the pixels within this box are considered independently moving.

E. Lane Detection and Filtering

A large number of lane detection algorithms has been reported which can roughly be divided into feature- and model-based methods, see e.g., [2]. The former detect street lanes

bottom-up, i.e., based on certain low-level features like intensity gradients, edges, color etc. e.g., [43], whereas the latter aim at identifying image parts corresponding to a predefined lane or street model, e.g., [44], [45]. Both approaches have known advantages and disadvantages, feature-based methods can detect arbitrary shapes but might add erroneously detected image parts into the returned street detection. The more restricted model-based methods are more robust to disturbances like occlusions, noise and shadows, however they are restricted to predefined lane shapes making them less flexible. To detect lanes in incoming image frames we employ a simple and fast (real-time, i.e., 20Hz) feature-based algorithm which works similar to contour tracers used in computer vision. First, edges and pixel orientations are computed by using standard computer vision techniques (Canny and Sobel operator, [46], [47]). Second, line segments are constructed by joining nearby edge pixels with similar orientations. Then, nearby line segments are further joined resulting in longer lines. Thus, a list of lines which might contain small interruptions is obtained. The parametrization of a lane can be seen from Fig. 7a.

The left and right lane are expected to be the longest of these lines starting in a particular area at the bottom of the image. During initialization this area is determined manually, and further tracked using a Kalman filter [48]. This is very simple and requires almost no a priori knowledge or initial image preprocessing, i.e., it works on raw intensity images containing street lanes from a single camera. To correct against false positives we apply an additional filter which uses 3D information to verify if the detected lane is on the ground plane. This is achieved by using the computed dense stereo map (see Section III-C) which attaches 3D information to each point of the extracted lane in form of disparity values. There is a linear relationship between disparity values and horizontal image coordinates and the filter checks whether the extracted lane fulfills this criterion. Since the disparity contains noise, we use RANSAC [49] to fit a line to the disparity data of the extracted lane. If an estimate with slope in a tolerable range can be fitted, we believe that the lane is on the ground plane, otherwise it is rejected.

As indicated by the entry "generate lane expectation" in Fig. 3 a final feedback mechanism aids the stability of the lane detection by generating lane expectations based on the human behavior. The velocity and steering angle of the car is used to derive its Rigid Body Motion (RBM), which is then used to predict the position of a detected lane one frame ahead.

The expected lane is then used for filtering the lane detection in the following frame by comparing both and rejecting the detection if it differs too much from the prediction.

F. The Perception-Action Repository

The PAR serves as the system's memory. It stores its (driving) experience and retrieves it at a later stage. The idea is based on the assumption that a human executes a stereotypic driving behavior according to the street trajectory he or she sees in front. A straight street ahead will be followed by straight steering for a while and probably some acceleration, a sharp turn will cause the driver to decelerate and to steer

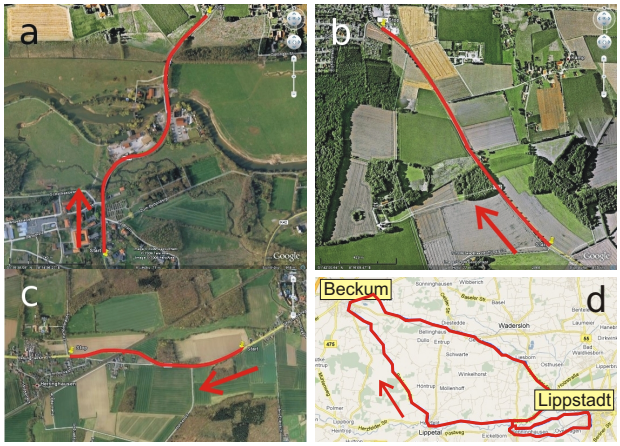


Fig. 6: The tracks on which training data was recorded, (a) s03 (1km), (b) s05 (2km), (c) s06 (1km), (d) long tour between Lippstadt and Beckum (69km).

accordingly, etc. To let the system learn this, we store a description of the street ahead together with *sequences* of human driving data that he or she issued after having observed that street. This is the training phase. After this follows the retrieval phase during which the system can use incoming street trajectory descriptions to query the PAR (similar to a pattern matching process) and obtain adequate driving sequences. The fact that we use sequences instead of single step actions allows us in a subsequent step to compute an expected human driving behavior that reaches to some extent into the future, i.e. we predict future sequences of human driving actions. To demonstrate the system's use for driver assistance we issue warnings if the actual human driving data differs too much from the computed expected driving. In the following we explain what data we use and then formalize the individual steps.

1) Data, Default Driving and Preprocessing: We use a data set from a single driver recorded and provided by Hella KGaA Hueck & Co[2]. It consists of three repeatedly driven tours to which we refer as s03, s05 and s06, comp. Fig. 6a-c, and a track that we denote "long tour", see Fig. 6d, which was driven twice.

The data predicted is steering and acceleration from the car CAN-bus, see Section III-A. Due to trends and a too high variance found in general in the velocity data we used the acceleration signal which we found to be better predictable.

The goal is to learn the default human behavior concerning steering and acceleration for lane following in context-free situations. By this we mean driving without additional traffic and without special situations, like intersections, etc. This requires filtering the original driving data to clean it from context-afflicted situations, which can be achieved by using the IMO detector to identify and exclude situations with IMOs in front as well as by using inconsistencies in the lane detection to identify and exclude situations like intersections. Hence, PAR entries for lane following are based on context-free driving examples.

[2]Parts of this set are available at the web-page [50].

In addition to the acquisition of context-free situations we also remove action sequences that are obvious outliers as described in the following. In Fig. 7b and c fifteen steering and acceleration signals from the same track and driver are shown along with their means plotted in black. We observe a much higher variance in the acceleration data than for steering which is quantified by the mean signal to noise ratio, SNR (we compute $E[\frac{|\mu|}{\sigma}]$, with μ being the mean and σ the standard deviation), where a high SNR indicates good and a low one bad predictability of a signal. For the shown data we obtain an SNR value for steering of 7.43 and for acceleration 0.62. By removing outliers (which are detected by an automatic procedure during which the individual signals are compared to the mean) the latter can be increased to 1.3. In Fig. 7d black signals are those acceleration signals found to be sufficiently similar and others (differently colored plots) are the ones that were sorted out. The sequence closest to the mean is plotted in red. This data is what we use for training the PAR, as will be explained in Section III-F2.

Driving data assigned to similar situations is averaged during training. Similarity is defined by the resemblance between left and right street lane of the observed image and one already contained in the PAR as well as a short sequence of previous steering data. A formal definition is given in Section III-F3. Hence, for tracks that are being driven multiple times the system defines the default human driving behavior over the mean of the action sequences obtained from driving on the same track as exemplary shown by the black plot in Fig. 7b. Note, however, learning starts with the first entry in the PAR and every additional information just improves the performance of the perception-action mapping. Single-example performance is reliable for steering, but remains restricted for acceleration control, see Section V. Note, as curve shapes are fairly similar, after some learning the system is able to generalize into unseen curves. This will be shown later (see Fig. 11).

Since the signals predicted by the PAR should correspond to the default behavior of the human, i.e., the mean action-signal, we evaluate the performance of the PAR by comparing its output against the human action sequence closest to the mean signal, i.e., the red plot in Fig. 7d, (which we do not include in the training data.)

2) PAR Training: The information stored as experience in the PAR is a state vector, $\mathbf{s}$, containing a description of extracted left and right street lane ($\mathbf{v}_{\text{left}}$, $\mathbf{v}_{\text{right}}$) from a current image frame, I_t. Here, t denotes the time of the image observation[3]. Because we found that only a description of the street ahead is not sufficient to distinguish different driving situations from each other, e.g., a straight street can lead to acceleration or no acceleration depending on the current velocity, we also store a short sequence of previous steering ($\mathbf{s}_{\text{past}}$). We use a fixed number of $m = 50$ discrete steering actions that preceded I_t. Thus, the state vector is as given in eq. 1. To each such state vector we assign sequences of future human driving actions executed after the observation of I_t,

[3]Time is measured in discrete time steps according to the camera's image capturing frequency.

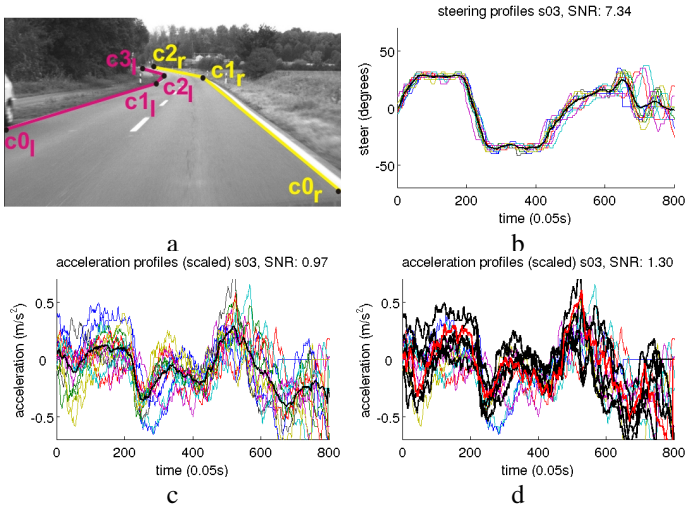


Fig. 7: (a) The extracted street boundaries described by polylines and the corner points. The vectors of the corner points $(c_{0_{l/r}}, c_{1_{l/r}} \dots)$ for left and right lane constitute the visual state description. (b) Steering and (c) acceleration signals from fifteen runs from the same track and driver. The mean is plotted in black. "SNR" refers to signal to noise ratio. (d) The sequence closest to the mean is shown in red. Black signals are considered to be sufficiently similar.

which we refer to as $\mathbf{s}_{\text{fut}}$ and $\mathbf{a}_{\text{fut}}$. The length of the sequences stored is supposed to resemble the number of actions necessary to cover the part of the street observable in I_t. Since we do not know exactly how many actions this corresponds to, we use a fixed value, $n = 100$, which is 5 seconds of driving corresponding to 97m at a speed of 70km/h which we consider reasonable for country road driving. Thus, a PAR entry, $\mathbf{e}$, is as given in eq. 2.

$$\mathbf{s} = \{\mathbf{v}_{\text{left}}, \mathbf{v}_{\text{right}}, \mathbf{s}_{\text{past}}\}, \qquad \text{state vector} \tag{1}$$

$$\mathbf{e} = \{\mathbf{s}, \mathbf{s}_{\text{fut}}, \mathbf{a}_{\text{fut}}\} \qquad \text{PAR entry} \tag{2}$$

The action sequences $\mathbf{s}_{\text{past/fut}}$ and $\mathbf{a}_{\text{fut}}$

$$\mathbf{s}_{\text{past}} = [s_{t-1}, s_{t-2}, \dots, s_{t-m}] \tag{3}$$

$$\mathbf{s}_{\text{fut}} = [s_t, s_{t+1}, \dots, s_{t+n}] \tag{4}$$

$$\mathbf{a}_{\text{fut}} = [a_t, a_{t+1}, \dots, a_{t+n}] \tag{5}$$

with s and a denoting single steering and acceleration signal values (actions).

The descriptions of the left and right street lane ($\mathbf{v}_{\text{left/right}}$) are linear approximations (polylines) of the extracted lanes in image coordinates, obtained by applying the Douglas-Peucker method [51]. We store this as vectors containing the corner points of the polylines as visualized in Fig. 7a, thus:

$$\mathbf{v}_{\text{right}} = [c_{0_r}, c_{1_r}, \dots, c_{l_r}] \tag{6}$$

$$\mathbf{v}_{\text{left}} = [c_{0_l}, c_{1_l}, \dots, c_{l_l}], \tag{7}$$

with l_l and l_r denoting the lengths of $\mathbf{v}_{\text{left}}$ and $\mathbf{v}_{\text{right}}$.

During the training phase a new entry, $\mathbf{e}$ (comp. eq. 2), is added to the PAR if it represents new knowledge, i.e., if no entries are already available containing similar information.

Precisely an entry is added if: a) there is no other entry already in the PAR with $\mathbf{v}_{\text{left}}$ and $\mathbf{v}_{\text{right}}$ having the same lengths as those contained in the probed entry, in other words if a completely new street trajectory is observed, or b) if there are such entries, but none of them have a state vector similar to the one probed for adding. Two state vectors are similar if the differences between their components are each below a fixed threshold, i.e., if eq. 8 and 9 are fulfilled:

$$\epsilon\_\mathrm{v} \leq thresh\_\mathrm{v} \tag{8}$$

$$\epsilon\_\mathrm{s}_{\text{past}} \leq thresh\_\mathrm{s}_{\text{past}} \tag{9}$$

where $\epsilon\_\mathrm{s}_{\text{past}}$ is the summed, squared difference between the entries of $\mathbf{s}_{\text{past}}$ of two state vectors, and $\epsilon\_\mathrm{v} = \epsilon\_\mathrm{v}_l + \epsilon\_\mathrm{v}_r$, and $\epsilon\_\mathrm{v}_{l/r}$ are the sums of the normalized, weighted and summed euclidean differences between $\mathbf{v}_{\text{left}}$ and $\mathbf{v}_{\text{right}}$ of two state vectors, see eq. 10.

$$\epsilon\_\mathrm{v}_{l/r} = \frac{1}{l_{l/r}} \sum_{i=0}^{l_{l/r}} \omega[i] \sqrt{(\mathrm{v}[i]_{\text{left/right}} - \mathrm{v}^*[i]_{\text{left/right}})^2}, \tag{10}$$

where the star in $\mathrm{v}^*[i]_{\text{left/right}}$ indicates that it is an entry of another state vector, and ω is a weight vector with $\omega[i] \geq \omega[i+1]$. The weighting punishes differences between two lanes close to the image bottom more than differences appearing closer to the horizon.

The PAR is complete if a predefined number of entries is reached, or the two cases in which entries are added to the PAR as just described do not occur anymore. Note, training is done off-line (hence, not during driving). This would be desired also in any commercial system as the training procedure is demanding (due to the removal of context-dependent scenes) and should thus take place when the car is not operating.

3) PAR Retrieval: To retrieve information from the PAR it is queried with a current state vector which is compared to all PAR entries whose $\mathbf{v}_{\text{left}}$ and $\mathbf{v}_{\text{right}}$ have the same lengths as those in the queried state vector. The action sequences attached to the most similar PAR entry are returned. Similarity is defined by computing the differences between the single entries of two state vector entries, i.e., ϵ_v, ϵ_s_{past}, as defined above, and the most similar entry is the one with the lowest overall differences.

Thus, the return parameters of a query are either: 1) the differences ϵ_v and ϵ_s_{past} between the most similar PAR entry and the query, and the action sequences $\mathbf{s}_{\text{fut}}$ and $\mathbf{a}_{\text{fut}}$ assigned to the most similar entry, or: 2) an indication that no match could be retrieved. The latter occurs either when there was no entry that the query could be compared to, or the best found match was unacceptably bad, i.e., the assigned differences exceeded predefined thresholds.

4) Prediction of Action Sequences: Because the PAR is queried every time step, sequences of driving behavior more or less appropriate to the queried situation are obtained. The degree to which the returned actions correspond to the queried situation is indicated by the returned differences ϵ_v, ϵ_s_{past}.

Since it is unlikely that identical state vectors are obtained multiple times even on the same track, a mechanism for generalization is required. That is, the system must compute adequate driving actions based on the more or less appropriate PAR returns. We postpone this step until retrieval time as typical for lazy-learning algorithms ([32], [52]–[54]) which are often used in imitation learning (compare [15]). The final expected human driving sequences are generated by keeping the latest k query results for steering and acceleration, and simply averaging over values belonging to the same time step (gray box in Fig. 8). Assuming that these values are contained in a buffer $\mathbf{g}_{\text{buf}}$, see Fig. 8, a single predicted action is computed as given in eq. 11.

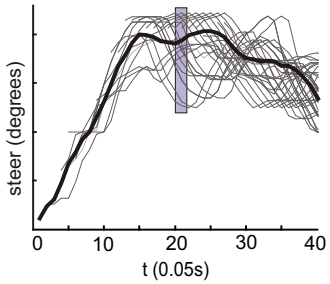


Fig. 8: Gray lines are PAR returns for steer and the computed expected human driving sequence is drawn in black. The gray rectangle indicates a vector $\mathbf{g}_{\text{buf}}$ containing all action signals of a certain time step used for averaging, see text.

$$a_t = \frac{1}{|\mathbf{g}_{\text{buf}}|} \sum_{i=0}^{|\mathbf{g}_{\text{buf}}|-1} \mathbf{g}_{\text{buf}}[i]. \tag{11}$$

Thus, every action command in the resulting prediction is a linear interpolation between the examples learned before. This is similar to the k-nearest neighbor algorithm which uses k closest training examples to compute a value for the variable of interest. Thus, the learning begins to work already with one single entry in the PAR and improves on repeatedly seen tracks just as a human driver would, too. For a final smoothing we apply a moving average filter (windowsize=10) on the resulting signal.

We implemented a simple warning mechanism to the system to demonstrate its use for driver assistance. A warning is generated if the actual human steering deviates too much from the expected driving given by the computed prediction and specified by a threshold determined empirically based on the off-line analysis of the test-drive sequences.

Fig. 4 exemplary shows the actual driving, the prediction and the thresholds.

G. Algorithmic Flow

The algorithmic flow of the system concerning action prediction is summarized in Fig. 9.

No output can be generated if lanes could repeatedly not be detected, or if they were repeatedly mis-detected, i.e., other items erroneously identified as lane. Some critical cases can be compensated by other mechanisms, e.g., the generated lane expectation can be used in case of an undetected lane. If this prediction is considered unreliable the previous action plan can be used for as long as it contains entries. In this case the control is open loop and only single action commands can be

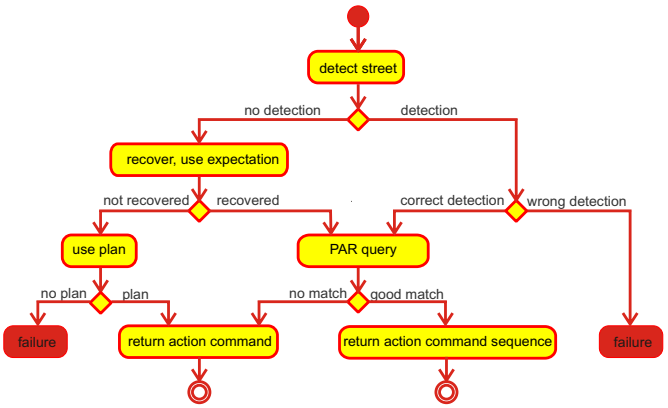


Fig. 9: Algorithmic flow of the driving system (UML activity diagram).

issued. In the optimal case the system returns a sequence of action commands as described in Section III-F4.

IV. Results

The performance of the IMO detection is reported in detail in [39], thus here we focus on the lane detection and the learning system.

The developed lane detection algorithm was evaluated on each tour of the available data set which comprised a variety of country roads with both clear and damaged lane markers (a total of 11.400 frames with humanly detectable lane markers present in 11.055 frames). In 98.9% of the 11.055 frames a valid lane description was extracted. In 5.6% of these cases only the left and in 8.5% only the right marker was detected. Both markers were found in 84.8% of these cases.

To evaluate how well the action predictions match the human default behavior we use the provided data set.

After filtering it, as explained in Section III-F1, approximately 80 min. of training data were obtained, resulting in a PAR with 90,470 entries, adequate for evaluating the system performance. First, we test the performance on a known track, i.e., one that the system had seen before. For that we train the PAR with all runs but the one closest to the mean which we consider to resemble the human default behavior as explained in Section III-F1 and which we use for testing. The smoothed steering and acceleration signals from the algorithm are plotted against the signal generated by the human for s03, s05, and s06 in Fig. 10. As an additional measure of similarity we compute the correlation coefficient of the two signals (human and prediction). For steering and acceleration prediction we obtain for s03 0.99 and 0.81, for s05 0.93 and 0.73, and for s06 0.97 and 0.67. Thus, all predictions are very good, however the steering signal is better approximated than acceleration, which is as expected from the computed SNR in Section III-F1.

For testing the performance on an unknown track we train the PAR with all available data except that from the track we test for. The result is shown in Fig. 11. We chose s05, which contains a very sharp turn (see Fig. 12) leading to two typical phenomena discussed below. The resulting steer and acceleration predictions in comparison to the human signal are

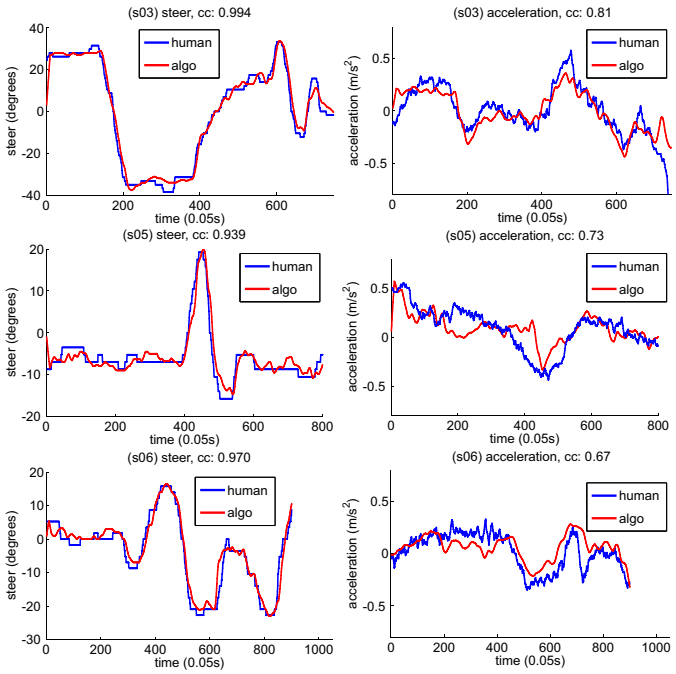


Fig. 10: Results for steering (left column) and acceleration (right column) prediction for known tracks. "cc" denotes the correlation coefficient value between the human generated signal and the synthesized one.

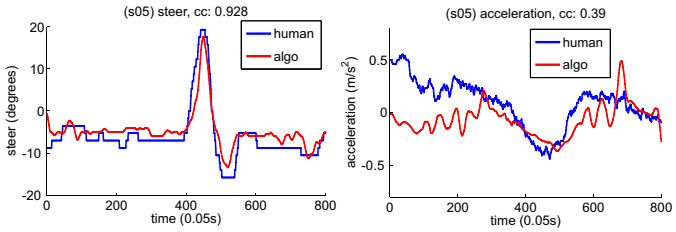
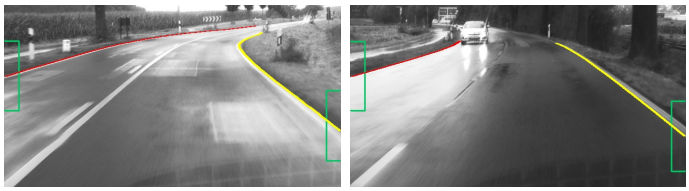


Fig. 11: Results for steering (left) and acceleration (right) prediction on an unknown track.



a b

Fig. 12: (a) Entering a sharp turn in s05 at $t = 200$. (b) Too short detected lanes.

shown in Fig. 11. The steering prediction is very close to the human signal, but the acceleration is less reliable. In particular it can be seen from the plotted acceleration prediction in Fig. 11 that the sharp curve (which is entered around time step 200, comp. Fig. 12a, is accompanied by a deceleration in the human signal (between time step 220 and 380) and that this is reflected nicely by the algorithm. However, before and after the deceleration part the predicted signal differs considerably from the human data.

This phenomenon results from the fact that street parts with a lower curvature allow a great variance in the driver's choice of speed depending on hard-to-access variables including "mood" and "intention", etc. where curves, especially sharp curves, significantly restrict the driver's behavior and thus make it better predictable. We therefore conclude that acceleration prediction with the presented method is only useful for curve negotiation. The second observation is that there are unwanted peaks in the predicted acceleration curve. This happens because the system sometimes hooks on to a different, similar looking road segments in the PAR, as can be seen from Fig. 12b. Although the lanes are correctly extracted the lookahead is not sufficient to distinguish this situation properly from almost straight streets. We found that our driver sometimes reacted to upcoming curves up to 8 seconds before entering them, requiring the system to have a similar lookahead to correctly predict the driver. Humans can see this far and make out even very faint features of the road layout, which leads to such early reactions, computer vision systems, however, cannot. The detected lanes at such distances and the detected lane segments will, for smooth and less descriptive situations, remain ambiguous leading to false matches which causes these peaks.

According to the algorithmic flow shown in Fig. 9 critical cases are a frequently undetected street, as well as the case that the PAR did not contain a good enough match. In these cases it is possible to work off a previously generated action plan, but only as long such a plan is available. For the presented tests the street detection rates were high: for s03 100%, for s05 96%, and for s06 98%. However, the case that no PAR match was retrieved occurred relatively frequently: for s03 in 32%, for s05 in 12%, and for s06 in 39% of all cases. For testing the case of driving on an unknown street on s05 no PAR match was retrieved in 39%. Despite these high rates it can be seen from the Figs. 10 and 11, that for the entire duration action signals were reliably produced. It is an important aspect of the system that its ability to predict sequences adds considerable robustness to its performance.

During the final review meeting of the DRIVSCO project the system was demonstrated successfully observed by three independent international reviewers, see [55]. The driver from which the training data was obtained, i.e., who taught the system, drove the first 20 minutes of the long tour and back, where "back" corresponds to an unknown track situation as curves are reverted. All components were shown to work reliably together at the desired speed, the lane detection worked even under challenging weather conditions where sunshine after rain caused reflections. (A TV report of this is available online [56].)

V. Discussion and Conclusion

We have presented the DRIVSCO system, which learns basic human driving (steering and acceleration control for lane following) by observing the driver. We evaluated its imitation performance on known and unknown tracks and showed it to work reliably for steering and, in the case of curve negotiation, also for acceleration prediction. In addition we have implemented a novel form of data driven IMO detection. Thus both, the learning algorithm as well as the IMO detection, do not rely on predefined models which makes the system widely applicable.

We demonstrated the use of the system output (steering and IMO detection) for supporting the driver. A domain in which this system may have future potential, is that of intelligent driver assistance systems which automatically adapt to individual driver behavior.

In contrast to most related work, DRIVSCO is an integrated system implemented on a real car, in real-time, realized by a multi-threaded, parallel CPU/GPU architecture which uses data from real driving – not simulation. The learning algorithm is deliberately simple and thus easy to understand and implement. In the early stages of the project different methods based on feed-forward and radial-basis-function networks were tested, however not achieving the performance of the lazy learning approach presented here. Furthermore, lazy learning offers the advantage that all information remains human interpretable, which is convenient for error analysis as opposed to learning algorithms which transform information into subsymbolic knowledge which is, for example, the case with neural networks. The system predicts *sequences* of expected human behavior as opposed to a moment-to-moment control. This makes it a) more stable, e.g., in case of unreliable or lacking sensory input it can use predictions as a fall-back plan, and b) it allows for proactive control, i.e., warnings can be issued based on the predicted behavior instead of the current one.

The system has been specifically designed for motorways and country roads, hence driving situations with limited context. To apply imitation learning to more difficult driving situations (e.g., city) appears currently infeasible as driving decisions are in these cases far too diverse and state-action descriptions would become too complex. Furthermore, we observed that acceleration signals are very non-descriptive when driving in uncritical situations (e.g., straight road) because drivers follow their mood. This contributes strongly to the high variance observed in the acceleration data. As a consequence, longitudinal control (or warning) becomes only useful whenever a driver is forced to drive with less leeway (e.g., in front of sharp curves). This notion is important when considering the psychological acceptance of individualized driving aids. One of their central features must be to not unnecessarily interfere with the driver. Hence, in uncritical situations systems should remain silent and acceleration should remain in the hands of the driver.

To improve the presented system one should furthermore consider to extend the system's lookahead, beyond that of machine vision. This could be achieved, for example, by integrating GPS information and digital maps.

One interesting aspect concerning industrial applications is the potential use of this system for night driving support. Under bad illumination conditions the human's sensing process is obviously limited, however, by using infrared light the system's sensing is less affected, given that the lane detection process is adequately adapted to the new circumstances. Thus, the system can use its knowledge about driving acquired

during the day to support the human in the more difficult situation of night driving.

REFERENCES

[1] S. Mammar, S. Glaser, and M. Netto, "Time to line crossing for lane departure avoidance: a theoretical study and an experimental setting," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 7, no. 2, pp. 226–241, June 2006.

[2] J. McCall and M. Trivedi, "Video-based lane estimation and tracking for driver assistance: survey, system, and evaluation," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 7, no. 1, pp. 20–37, March 2006.

[3] (2008) collected publications of the safelane project. [Online]. Available: http://www.preventip.org/en/public_documents/publications/safelane_publications.htm

[4] L. Li, F.-Y. Wang, and Q. Zhou, "Integrated longitudinal and lateral tire/road friction modeling and monitoring for vehicle motion control," vol. 7, no. 1, March 2006, pp. 1–19.

[5] P. Ulleberg and T. Rundmo, *Safety Science*, vol. 41, no. 5, pp. 427 – 443, 2003. [Online]. Available: http://www.sciencedirect.com/science/article/B6VF9-487KCBW-4/2/b09939b6c416cac31aa9979e6eb6c60a

[6] F. I. Kandil, A. Rotter, and M. Lappe, "Driving is smoother and more stable when using the tangent point," *Journal of Vision*, vol. 9, pp. 1–11, 2009.

[7] K. A. Brookhuis, D. de Waard, and W. H.Janssen, "Behavioural impacts of advanced driver assistance systems-an overview," *European Journal of Transport and Infrastructure Research*, vol. 1, no. 3, 2001.

[8] A. Lindgren and F. Chen, "State of the art analysis: An overview of advanced driver assistance systems (adas) and possible human factors issues," in *Human Factors and Economic Aspects on Safety. Swedish Network for Human Factors Conference*, 2007.

[9] J. F. Coughlin, B. Reimer, and B. M. (2009)., "Driver wellness, safety & the development of an awarecar," AgeLab, MIT, White Paper, 2009.

[10] S. Hoch, M. Schweigert, F. Althoff, and G. Rigoll, "The bmw surf project: A contribution to the research on cognitive vehicles," in *Proceedings of the 2007 Intelligent Vehicles Symposium*, 2007.

[11] E. D. Dickmanns and V. Graefe, "Dynamic monocular machine vision," *Machine Vision and Applications*, vol. 1, pp. 223–240, 1988.

[12] M. A. Turk, D. G. Morgenthaler, K. D. Gremban, and M. Marra, "Vits-a vision system for autonomous land vehicle navigation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 10, no. 3, pp. 342–361, 1988.

[13] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, K. Lau, C. Oakley, M. Palatucci, V. Pratt, P. Stang, S. Strohband, C. Dupont, L. E. Jendrossek, C. Koelen, C. Markey, C. Rummel, J. van Niekerk, E. Jensen, P. Alessandrini, G. Bradski, B. Davies, S. Ettinger, A. Kaehler, A. Nefian, and P. Mahoney, "Stanley: The robot that won the darpa grand challenge:," *J. Robot. Syst.*, vol. 23, no. 9, pp. 661–692, 2006. [Online]. Available: http://dx.doi.org/10.1002/rob.v23:9

[14] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, J. Dolan, D. Duggins, D. Ferguson, T. Galatali, C. Geyer, M. Gittleman, S. Harbaugh, M. Hebert, T. Howard, A. Kelly, D. Kohanbash, M. Likhachev, N. Miller, K. Peterson, R. Rajkumar, P. Rybski, B. Salesky, S. Scherer, Y. Woo-Seo, R. Simmons, S. Singh, J. Snider, A. Stentz, W. . Whittaker, and J. Ziglar, "Tartan racing: A multi-modal approach to the darpa urban challenge," *Darpa Technical Report*, 2007.

[15] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robot. Auton. Syst.*, vol. 57, no. 5, pp. 469–483, 2009.

[16] D. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," in *Advances in Neural Information Processing Systems 1*. Morgan Kaufmann, 1989.

[17] ——, "Neural network based autonomous navigation," in *NAVLAB90*, 1990, pp. 558–614.

[18] ——, "Efficient training of artificial neural networks for autonomous navigation," *Neural Computation*, vol. 3, no. 1, pp. 88–97, 1991.

[19] D. A. Pomerleau, "Neural network vision for robot driving," in *The Handbook of Brain Theory and Neural Networks*. M. Arbib, 1999.

[20] M. Pasquier and R. J. Oentaryo, "Learning to drive the human way: a step towards intelligent vehicle," *International Journal of Vehicle Autonomous Systems*, vol. 6, pp. 24–47(24), December 2007. [Online]. Available: http://dx.doi.org/10.1504/IJVAS.2008.016477

[21] C. Sammut, S. Hurst, D. Kedzier, and D. Michie, "Learning to fly," in *ML*, 1992, pp. 385–393.

[22] A. Y. Ng and S. Russel, "Algorithms for inverse reinforcement learning," in *Proc. ICML*, 2000.

[23] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *In Proceedings of the Twenty-first International Conference on Machine Learning*. ACM Press, 2004.

[24] (2010) Official diplecs website. [Online]. Available: http://www.diplecs.eu/

[25] C. H. Hwang, N. Massey, B. W. Miller, and K. Torkkola, "Hybrid intelligence for driver assistance," in *FLAIRS*, 2003.

[26] R. Freymann, *Motion and Vibration Control*. Springer Netherlands, 2008, ch. Driver Assistance Technology to Enhance Traffic Safety, pp. 71–81.

[27] K. Brookhuis and D. de Waard, "Limiting speed, towards an intelligent speed adapter (isa),," *Transportation Research Part F: Traffic Psychology and Behaviour*, vol. 2, pp. 81–90, 1999.

[28] A. Tahirovic, S. Konjicija, Z. Avdagic, G. Meier, and C. Wurmthaler, "Longitudinal vehicle guidance using neural networks," in *Computational Intelligence in Robotics and Automation, 2005. CIRA 2005.*, 2005.

[29] D. Partouche, M. Pasquier, and A. Spalanzani, "Intelligent speed adaptation using a self-organizing neuro-fuzzy controller," in *Proc. IEEE Intelligent Vehicles Symposium*, 2007, pp. 846–851.

[30] H. Kwasnicka and M. Dudala, "Neuro-fuzzy driver learning from real driving observations," in *Proceedings of the Artificial Intelligence in Control and Managamnent*, 2002.

[31] I. Markelic, T. Kulvicius, M. Tamosiunaite, and F. Wörgötter, "Anticipatory driving for a robot-car based on supervised learning," in *ABiALS*, 2008, pp. 267–282.

[32] D. W. Aha, Ed., *Lazy Learning*, ser. Artificial Intelligence Review. Kluwer Academic Publishers, 1997, vol. 11, ch. Editorial, pp. 7–10.

[33] J. Daugman, "Uncertainty relation for resolution in space, spatial-frequency, and orientation optimized by two-dimensional visual cortical filters," *J. Opt. Soc. Am. A-Opt. Image Sci. Vis.*, vol. 2, no. 7, pp. 1160–1169, 1985.

[34] K. Pauwels and M. Van Hulle, "Realtime phase-based optical flow on the GPU," in *IEEE Conference on Computer Vision and Pattern Recognition, Workshop on Computer Vision on the GPU*, 2008.

[35] T. Gautama and M. Van Hulle, "A phase-based approach to the estimation of the optical flow field using spatial filtering," *IEEE Transactions on Neural Networks*, vol. 13, no. 5, pp. 1127–1136, 2002.

[36] S. Sabatini, G. Gastaldi, F. Solari, J. Diaz, E. Ros, K. Pauwels, M. Van Hulle, N. Pugeault, and N. Krüger, "Compact and accurate early vision processing in the harmonic space," in *International Conference on Computer Vision Theory and Applications*, Barcelona, 2007, pp. 213–220.

[37] B. Leibe, K. Schindler, N. Cornelis, and L. Van Gool, "Coupled object detection and tracking from static cameras and moving vehicles," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 10, pp. 1683–1698, 2008.

[38] B. Leibe, N. Cornelis, K. Cornelis, and L. Van Gool, "Dynamic 3d scene analysis from a moving vehicle," in *IEEE Conference on Computer Vision and Pattern Recognition*, Minneapolis, 2007, pp. 1423–30.

[39] K. Pauwels, N. Krueger, M. Lappe, F. Woergoetter, and M. van Hulle, "A cortical architecture on parallel hardware for motion processing in real-time," *Journal of Vision*, submitted.

[40] T. Zhang and C. Tomasi, "On the consistency of instantaneous rigid motion estimation," *International Journal of Computer Vision*, vol. 46, pp. 51–79, 2002.

[41] F. Mosteller and J. Tukey, *Data Analysis and Regression: A Second Course in Statistics*. Mass.: Addison-Wesley Reading, 1977.

[42] W. Thompson and T. Pong, "Detecting moving-objects," *International Journal of Computer Vision*, vol. 4, pp. 39–57, 1990.

[43] M. Bertozzi and A. Broggi, "Real-time lane and obstacle detection on the system," in *IEEE Intelligent Vehicles, 1996*, 1996, pp. 213–218.

[44] E. Dickmanns, *Dynamic Vision for Perception and Control of Motion*. Springer, 2007.

[45] M. Aly, "Real time detection of lane markers in urban streets," in *Proc. IEEE Intelligent Vehicles Symposium*, 4–6 June 2008, pp. 7–12.

[46] J. F. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 8, pp. 679–698, 1986.

[47] I. Sobel and G. Feldman, *A 3x3 Isotropic Gradient Operator for Image Processing*. Wiley, 1973.

[48] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Transaction of the ASMEJournal of Basic Engineering*, pp. 33–45, 1960.

[49] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[50] (2010) Eisat website. [Online]. Available: http://www.mi.auckland.ac.nz/EISATS
[51] D. H. Douglas and T. K. Peucker, "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature," *Cartographica: The International Journal for Geographic Information and Geovisualization*, vol. 10, pp. 112–122, 1973.
[52] L. Bottou and V. Vapnik, "Local learning algorithms," *Neural Computation*, vol. 4, pp. 888–900, 1992.
[53] J. C. Santamaria, R. S. Sutton, and A. Ram, "Experiments with reinforcement learning in problems with continuous state and action spaces," *Adaptive Behavior*, vol. 6, pp. 163–217, 1997.
[54] W. D. Smart and L. P. Kaelbling, "Practical reinforcement learning in continuous spaces," in *Proceedings of the Seventeenth International Conference on Machine Learning*, 2000.
[55] "Future and emerging technologies (fet), newsletter," January 2010. [Online]. Available: ftp:ftp.cordis.europa.eupubfp7ictdocsfetfetnl06_en.pdf
[56] (2009) Tv report about the drivsco system (in german). [Online]. Available: http://www1.ndr.de/mediathek/index.html?media=ndsmag2340

Irene Markelić Irene Markelić is a doctoral student at the University of Göttingen in Germany. She received her B.Sc. and M.Sc. degree in computer science at the University of Koblenz-Landau in 2002 and 2005. Her research interests include vision based robotics and cognitive skill learning.

Anders Kjær-Nielsen received his B.Sc. and M.Sc. degree in Computer Systems Engineering from the University of Southern Denmark, Denmark, in 2004 and 2007 respectively. He is currently a PhD student at the Mærsk McKinney Møller Institute, University of Southern Denmark. His research interests include real-time processing, computer vision, embedded systems and FPGAs.

Karl Pauwels received the M.Sc. degree in Commercial Engineering, the M.Sc. degree in Artificial Intelligence, and the Ph.D. degree in Medical Sciences from the Katholieke Universiteit Leuven, Belgium. He is currently a postdoc at the Laboratorium voor Neuro- en Psychofysiologie, Medical School, K.U.Leuven. His research interests include dense optical flow, stereo and camera motion estimation, video stabilization, and real-time computer vision.

Lars Baunegaard With Jensen received his B.Sc. and M.Sc. degree in Computer Systems Engineering from the University of Southern Denmark, Denmark, in 2004 and 2007 respectively. He is currently a PhD student at the Mærsk McKinney Møller Institute, University of Southern Denmark. His research interests include real-time computer vision and GPU computing.

Nikolay Chumerin received the M.Sc. in Mathematics and Educational Science and Higher Educational Certificate of teacher in Mathematics and Computer Science from the Brest State University, Belarus in 1999. He is currently a PhD student at the Laboratorium voor Neuro- en Psychofysiologie, Medical School, K.U.Leuven. His research interests include biologically-inspired computer vision, machine learning and brain-computer interfacing (BCI).

Aušra Vidugiriene Aušra Vidugirien, received her B.Sc. and M.Sc. degree in computer science at Vytautas Magnus University (Kaunas) in Lithuania in 2003 and 2005 respectively, on language technologies. She is a doctoral student at Vytautas Magnus University (Kaunas) in Lithuania. Her research interests include signal processing and analysis of driver's bahavior.

Minija Tamosiunaite Minija Tamosiunaite has received her Engineer Diploma from Kaunas University of Technology, Lithuania, in 1991 and Ph.D. from Vytautas Magnus University, Lithuania, in 1997. Her research interests include machine learning and applications in robotics.

Alexander Rotter Alexander Rotter received his Diploma degree in electrical engineering and the Diploma degree in industrial engineering from Fachhochschule Sdwestfalen, Iserlohn, Germany in 2002 and 2007, respectively. He is currently working at Hella KGaA Hueck & Co. at the Advanced Development department. His research interests are vision based driver assistance systems, including object- and lane-detection, and emergency braking systems in combination with distance measurement sensors. Furthermore vision based driver assistance systems, object detection, and emergency braking systems in combination with distance measurement sensors.

Marc Van Hulle Prof. Dr. Marc Van Hulle is a Full Professor at the K.U.Leuven, Medical School, where he heads the Computational Neuroscience group of the Laboratorium voor Neuro- en Psychofysiologie. Marc Van Hulle received a M.Sc. degree in Electrotechnical Engineering (Electronics) and a Ph.D. in Applied Sciences from the K.U.Leuven, Leuven (Belgium). In 1992, he has been with the Brain and Cognitive Sciences department of the Massachusetts Institute of Technology (MIT), Boston (USA), as a postdoctoral scientist. In 2003, he received from Queen Margrethe II of Denmark the Doctor Technices degree, and in 2009 a Honory Doctoral degree from the Brest State University.

His research interests include computational neuroscience, neural networks, computer vision, data mining and signal processing.

Norbert Krüger is a Professor at the Mærsk McKinney Møller Institute, University of Southern Denmark. He holds a M.Sc. from the Ruhr-Universität Bochum, Germany and his Ph.D. from the University of Bielefeld. He is a partner in several EU and national projects: PACO-PLUS, Drivsco, NISA, Handyman.

Norbert Krüger is leading the Cognitive Vision Lab which is focussing on computer vision and cognitive systems, in particular the learning of object representations in the context of grasping. He has also been working in the areas of computational neuroscience and machine learning.

Florentin Wörgötter Florentin Wörgötter has studied Biology and Mathematics in Düsseldorf. He received his PhD in 1988 in Essen working experimentally on the visual cortex before he turned to computational issues at the Caltech, USA (1988-1990). After 1990 he was researcher at the University of Bochum concerned with experimental and computational neuroscience of the visual system. Between 2000 and 2005 he had been Professor for Computational Neuroscience at the Psychology Department of the University of Stirling, Scotland where his interests strongly turned towards "Learning in Neurons". Since July 2005 he leads the Department for Computational Neuroscience at the Bernstein Center at the University of Göttingen. His main research interest is information processing in closed-loop perception-action systems, which includes aspects of sensory processing, motor control and learning/plasticity. These approaches are tested in walking as well as driving robotic implementations. His group has developed the RunBot a fast and adaptive biped walking robot.